Bluetooth Adaptive Techniques to Mitigate Interference

N. Golmie and O. Rebala
National Institute of Standards and Technology
Gaithersburg, Maryland 20899
Email: nada.golmie@nist.gov

Abstract—**In this paper, we investigate the use of adaptive techniques to mitigate interference for Bluetooth systems in the presence of WLAN direct sequence spread spectrum devices. We consider two different techniques that attempt to avoid time and frequency collisions of WLAN and Bluetooth transmissions. We conduct a comparative analysis of their performance for several dynamic scenarios where the WLAN interference varies over time due to either change in user activity or number of non-overlapping WLAN systems. We discuss the trade-offs involved in terms of delay, packet loss performance, and synchronization.**

I. Introduction

Recently, there has been a growing number of industry led activities focused on the coexistence of wireless devices in the 2.4 GHz band. Both, the IEEE 802.15.2 Coexistence Task Group and the Bluetooth Special Interest Group (SIG) are looking at similar techniques for alleviating the impact of interference. The proposals considered by the groups range from collaborative schemes intended for Bluetooth and IEEE 802.11 protocols to be implemented in the same device to fully independent solutions that rely on interference detection and estimation. Except for a Time Division Multiple Access (TDMA) technique aimed at time sharing the Bluetooth and 802.11 signals [1], most mechanisms considered do not require any direct communication between the protocols. These so-called non-collaborative mechanisms range from adaptive frequency hopping [2] to packet scheduling and traffic control [3]. The techniques used for detecting the presence of other devices in the band are based on measuring the bit or frame error rate, and the signal to interference ratio. For example, each device can maintain a packet error rate measurement per frequency used. Frequency hopping devices can then know which frequencies are occupied by other users of the band and thus modify their frequency hopping pattern. They can even choose not to transmit on a certain frequency if that frequency is occupied. The first technique is known as adaptive frequency hopping, while the second technique is known as MAC scheduling or Bluetooth Interference Aware Scheduling (BIAS).

In this paper, we investigate the use of BIAS and an AFH technique. Although both techniques rely on similar methods for estimating the interference environment before a packet transmission, they differ signicantly in terms of complexity and performance. Our goal is to bring to light some of the trade-offs associated with different interference scenarios, applications, and parameters.

The remainder of this paper is organized as follows. In section II, we briefly describe BIAS and an AFH technique. In section III, we discuss the performance results obtained. In section IV, we offer concluding remarks.

II. Adaptive Interference Mitigation Techniques

Central to most interference mitigation techniques is the ability to detect the presence of other systems operating in the band. One method to estimate interference consists of measuring the percentage of packets dropped, Pr(Ploss), per frequency per receiver. Thus, given Pr(Ploss) and let's say a threshold value of 0.5, frequencies at each receiver are classified "good" or "bad" depending on whether their packet loss rate is less than or greater than 0.5 respectively. Also, updating Pr(Ploss) may vary according to the application, the environment, and the level of accuracy and interference tracking desired. In our simulations, we use a minimum update interval of 2 and a maximum of 100 seconds. Details on the dynamic procedure used to vary the update interval between this minimum and maximum values are found in [4].

Since in a Bluetooth piconet, the master device controls all packet transmission, the measurements collected by the slave devices are sent to the master (or implied in acknowledgement packets) that decides to (1) either avoid data transmission to a slave experiencing a "bad" frequency in the case of BIAS, and/or (2) modify the frequency hopping pattern in the case of AFH. While in the former case the decision remains local to the master, in the latter case, the master needs to communicate the changes in the hopping sequence to all slaves in the piconet in order to maintain synchronization. Thus, AFH requires the exchange of Layer Management Protocol (LMP) messages in order to advertise the new hopping sequence. Observe that this constitutes one of the major differences between BIAS and AFH.

A. Interference Aware Scheduling

The basic idea of the so-called Bluetooth Interference Aware Scheduling (BIAS) is that a data transmission to a slave experiencing a "bad" frequency is postponed until a "good" frequency is found in the hopping pattern. Furthermore, since a slave transmission always follows a master transmission, using the same principle, the master avoids receiving data on a "bad" frequency, by avoiding a transmission on a frequency preceding a "bad" one in the hopping pattern. Further details on BIAS are found in [5].

B. Adaptive Frequency Hopping

From the many AFH algorithms possible, we propose an AFH algorithm that modifies the original Bluetooth frequency hopping scheme as follows.

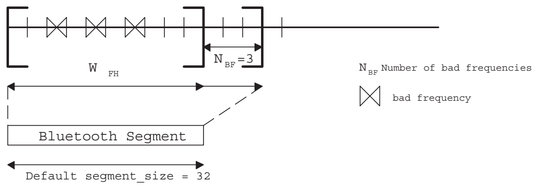

Fig. 1. Resizing the Frequency Hopping Window, W$_{FH}$

Given a segment of 32 “good” and “bad” frequencies that we call the Frequency Hopping Window (W$_{FH}$), the algorithm visits each “good” frequency exactly once. Each “bad” frequency in the segment is replaced with a “good” frequency selected from outside the original segment of 32 as shown in Figure 1. Thus, W$_{FH}$ is increased by the number of “bad” frequencies, N$_{BF}$, in the original segment. Thus, the difference between AFH and the original Bluetooth hopping sequence algorithm is in the selection of only “good” frequencies in order to fill up the segment size.

Finally, AFH does not preclude additional scheduling techniques to control the transmission (and possibly the retransmission) of packets on the medium.

III. Performance Evaluation

In this section we present simulation results to evaluate the performance of the proposed AFH algorithm. We ran several experiments using different applications, traffic types, and network topologies. Given the lack of space, only a representative set of the data obtained is discussed in this paper. While the advantages of AFH may be obvious in terms of mitigating the interference between Bluetooth and WLAN in a stationary environment, we focus mainly on its dynamic behavior and its ability to adjust to time-varying and different interference levels. Although, more difficult to solve, the scenarios selected are perhaps more realistic. A summary of the experiments is given in Table I.

TABLE I
Experiment Summary

Experiment	Traffic	Topology
1	ON/OFF Exponential	1
2	FTP/HTTP	1
3	ON/OFF Exponential	2 w/ 1 Bluetooth Piconet
4	ON/OFF Exponential	2 w/ 3 Bluetooth Piconets

A. *Experiment 1: Variable Offered Load*

We use Topology 1 illustrated in Figure 2 with one WLAN system (Access Point/Station) and a Bluetooth master/slave pair. The WLAN access point (AP) is located at (0,15) meters, and the WLAN station is fixed at (0,1) meters. The Bluetooth slave device is fixed at (0,0) meters and the master is fixed at (1,0) meters.

In Experiment 1, we use an on-off traffic generation model with parameters to characterize burstiness and user activity as seen at layer 2, such as packet interarrival, and packet size. For Bluetooth the packet size is fixed to either 1, or 5 slots using the

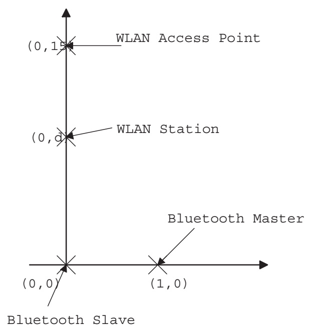

Fig. 2. Topology 1 - Two WLAN devices and one Bluetooth piconet

DH format and the mean interarrival time is computed according to

$$t_B = 2 \times n_s \times T_s/\lambda, \tag{1}$$

where λ is the offered load, n_s is the number of slots occupied by a packet. For $DH5$, $n_s = 5$. T_s is the slot size equal to 625 μs.

For WLAN, we fix the packet payload to $12,000$ bits assuming an IP packet of 1500 bytes and compute the mean packet interarrival time according to

$$t_W = (\frac{192}{1,000,000} + \frac{12,224}{data\_rate})/\lambda, \tag{2}$$

where λ is the offered load, 224 is the MAC layer header, 192 is the PLCP header (always sent at 1 Mbit/s). The payload is transmitted at the data_rate of 11 Mbit/s. The offered load for Bluetooth is varied between 10 and 100%, while the WLAN offered load is set to 60%. The time the WLAN connection is ON, T$_{ON}$, is exponentially distributed with a mean equal to 10 seconds, while the time the WLAN connection is OFF, T$_{OFF}$, is also exponentially distributed with mean equal to 20 seconds. Each simulation is run for 900 seconds. Averages and confidence intervals are obtained over 10 simulation runs.

Figure 3(a) gives the packet loss results at the Bluetooth slave. Observe that with AFH the packet loss is close to 8% and 4% for DH5 and DH1 packets respectively. These results are close to the ones obtained when no interference mitigation technique is used and depend on the frequency of the synchronization messages exchanged between the Bluetooth master and the slave. Thus there is a trade-off between the communication overhead and the response to changes in the interference environment. A fast responding system will incurr a lower packet loss at the cost of a higher communication overhead. In this experiment, synchronization messages are exchanged every 2 seconds, which is also the value used for E$_{min}$. On the other hand, using BIAS without AFH reduces the packet loss to less than 2% and 0.9% for DH5 and DH1 packets respectively at 10% offered load. As the offered load is increased, the packet loss even drops further to negligible levels. Since no explicit message exchange is required for BIAS, the response time to changes in the interference environment happen within a packet round trip time. Figure 3(b) illustrates the access delay results. For DH1 packets, AFH yields lower delays than BIAS. Delaying the transmission of a short packet in the case where the probability of packet collision is less than 22/79 leads to higher access delays. The access delay curve for AFH takes off at around

70% load. For DH5 packets, the delays obtained with AFH are comparable to the results obtained with BIAS loads less than 50%. However, as the offered load is increased the access delays obtained with BIAS are lower mainly due to fast response times and low packet loss.

B. Experiment 2: FTP and HTTP Profile

In this experiment, we use Topology 1 given in Figure 2. We consider two application profiles, namely, FTP and HTTP. We use the TCP/IP stack implemented in the OPNET library and configure the application parameters provided. For the FTP profile, the parameters are the percentage of put/get, the inter-request time, and the file size. The percentage of put/get represents the number of times the put command is executed in an FTP connection over the total number of put and get commands, i.e., a fifty percent indicates that half of the FTP commands executed are put, and the other half are get. The inter-request time is the interval between two FTP commands, and the file size represents the size of the file requested in bytes. The HTTP profile is described by parameters characterizing a web page such as the page interarrival time, the number of objects in each page and their size in bytes. Two profile sets are defined for each of the Bluetooth and the WLAN in Table II.

TABLE II
PROFILE PARAMETERS

Parameters	Distribution	Value
Bluetooth FTP		
Percentage of Put/Get		100%
Inter-Request Time (seconds)	Exponential	5
File Size (bytes)	Constant	2M
WLAN FTP		
Percentage of Put/Get		100%
Inter-Request Time (seconds)	Exponential	1
File Size (bytes)	Constant	2M
HTTP		
Page Interarrival Time (seconds)	Exponential	5
Number of Objects per page	Constant	2
Object 1 Size (bytes)	Constant	10K
Object 2 Size (bytes)	Uniform	(200K,600K)

We ran two simulations where in each case both the WLAN and Bluetooth devices are using either the HTTP or the FTP profile. Table III gives the packet loss using FTP and HTTP traffic. Observe that the packet loss with BIAS alone is an order of magnitude lower than with AFH due to the dynamic nature of the traffic. The packet loss with AFH is comparable to the packet loss obtained when no algorithm is used.

On the other hand, the FTP access delay with BIAS is slightly higher than with AFH and None in that order. This increase in delay can be attributed to the policy of delaying the transmission of packets on the so-called "bad" frequencies, although the probability that a packet collision occurs is less than 22/79 (in fact it is proportional to the offered load and the number of channels occupied by WLAN divided by the number of frequencies used by Bluetooth). There is no noticeable difference in delay for the HTTP application given that it does not generate as much traffic as the FTP application.

TABLE III
EXPERIMENT 2: BLUETOOTH PROBABILITY OF PACKET LOSS

	BIAS	AFH	None
FTP	0.005	0.037	0.059
HTTP	0.005	0.019	0.021

TABLE IV
EXPERIMENT 2: BLUETOOTH ACCESS DELAY (SECONDS)

	BIAS	AFH	None
FTP	0.0040	0.0033	0.0029
HTTP	0.0026	0.0023	0.0023

C. Experiment 3: Multi-WLAN Interference

In this experiment, our goal is to study the performance of AFH in a multi-WLAN environment, where the Bluetooth hopping sequence is close to the minimum number of hops allowed, which in our case is set to 15. We use Topology 2 illustrated in Figure 4, consisting of 3 WLAN systems (source-sink pairs). In this experiment (Experiment 3) we use one Bluetooth piconet, while in Experiment 4, we use all three Bluetooth piconets. We use the same traffic parameters described in Experiment 1. Figure 5 gives the packet loss and access delay measured at the Bluetooth slave. In this case, there are three WLAN systems occupying about 9-11 frequencies each. That leaves about 18-20 frequencies in the band to be used by Bluetooth. With BIAS, the Bluetooth piconets only transmits on "good" frequencies, and therefore has to skip approximately 1 in every 4 transmission opportunity. With AFH, the frequency hopping sequence is modified in order to include only "good" frequencies. Therefore, we expect significant throughput and delay improvements with AFH. Observe in Figure 5(a) the packet loss with AFH (DH5) is around 2%. It is negligible with AFH for DH1 packets, and BIAS for both DH1 and DH5 packets. As expected, the access delay with AFH, shown in Figure 5(b), is several orders of magnitude lower than with BIAS. The delay for DH1 packets stays around 1 ms until an offered load of 70%. The delay for DH5 packets starts at 10 ms with a steep slope between 10 and 60%. At 60% the delay reaches 100 ms. On the other hand, the delay with BIAS for DH1 packets is 15 ms at 10% load and quickly reaches 10 seconds at 20% load. For DH5 packets, the delay between 10 and 40% load is comparable to the delay obtained with AFH. However, the delay curve takes off sharply at 40%. The delay is around 10 seconds between 40 and 100% offered load.

D. Experiment 4: Multi-WLANs and Bluetooth Piconets

In this experiment we use Topology 2 illustrated in Figure 4, with all three Bluetooth piconets (two additional Bluetooth piconets to what was used in Experiment 3) to highlight the performance of AFH in an extreme case of interference where multi-Bluetooth and WLAN devices are operating in the same environment. Our goal is to verify that the performance improvements observed with AFH in the previous experiment still apply in this case.

Figure 6 gives the packet loss and access delay measured at the Bluetooth slaves and averaged over all three slaves. As expected in this case, the presence of two more Bluetooth piconets makes the interference detection more challenging. As a result the packet loss observed in Figure 6 is higher than in Figure 5. The packet loss for AFH (8%) is about an order of magnitude higher than with BIAS (around 1% and 0.8% for DH5 and DH1 packets respectively). The difference in packet loss between the two schemes is due using a reduced hopping set with AFH while increasing the number of Bluetooth piconets. Similarly, the access delay in Figure 6(b) is higher than the delay in Figure 5(b) for both schemes. The delay with AFH for DH1 packets is around 2 ms. On the other hand the delay for DH5 packets is between 20 and 100 ms for offered loads between 10 and 70%. The delay with BIAS for both types of packets is between 10 and 20 seconds.

IV. Concluding Remarks

In this paper, we study using adaptive frequency hopping for Bluetooth devices when operating in close proximity to WLAN systems. We present the details of an AFH algorithm and compared its performance to BIAS, a delay transmission method aimed at interference mitigation. A summary of our findings is as follows. In the case of WLAN interference, when the interference levels vary over time and the channel estimation is performed often, the main advantages of AFH in terms of lowering the access delay, applies only to short packets (DH1). BIAS is more effective for longer packets (DH5) and leads to lower packet loss and comparable access delays. In the case of multi-WLANs (three non-overlapping systems), AFH keeps the delay low and doubles the throughput obtained with BIAS. This result holds even in the presence of two additional Bluetooth piconets. Thus, significantly reducing the frequency hopping sequence due to WLAN interference, causes only a slight increase in the probability of packet collisions among the different Bluetooth piconets and does not affect performance.

References

[1] J. Lansford, A. Stephens, and R. Nevo, "Wi-Fi (802.11b) and Bluetooth: Enabling Coexistence," in *IEEE Network Magazine*, Sept/Oct. 2001, vol. 15, pp. 20–27.

[2] B. Treister, A. Batra, K.C. Chen, O. Eliezer, "Adapative Frequency Hopping: A Non-Collaborative Coexistence Mechanism," in *IEEE P802.11 Working Group Contribution, IEEE P802.15-01/252r0*, Orlando, FL, May 2001.

[3] Carla F. Chiasserini, and Ramesh R. Rao, " Coexistence mechanisms for interference mitigation between IEEE 802.11 WLANs and bluetooth ," in *Proceedings of INFOCOM 2002*, 2002, pp. 590–598.

[4] N. Golmie, O. Rebala, and N. Chevrollier, "Bluetooth Adaptive Frequency Hopping and Scheduling," in *Proceedings of MILCOM'03*, Boston, MA, October 2003.

[5] N. Golmie, "Bluetooth Dynamic Scheduling and Interference Mitigation," in *to appear in ACM Mobile Network, MONET*, 2003.

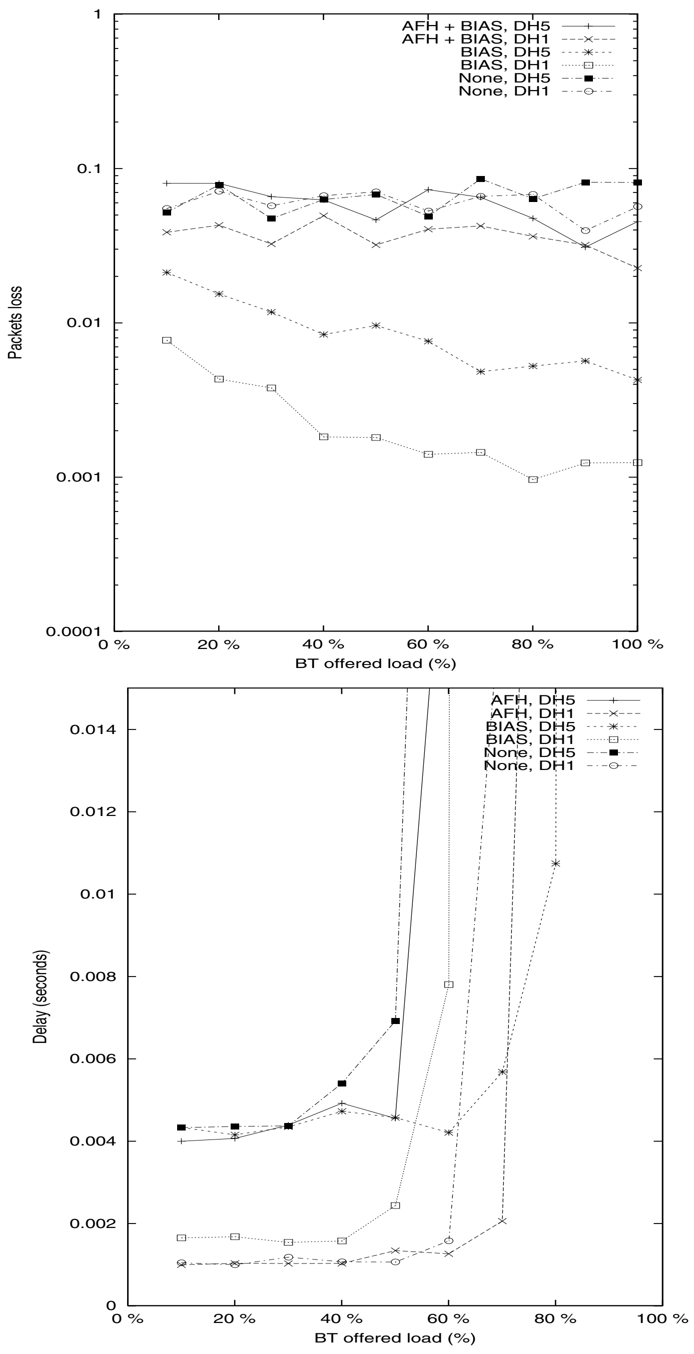

Fig. 3. $\boxed{\frac{(a)}{(b)}}$ Experiment 1. (a) Probability of Packet Loss. (b) Mean Access Delay (seconds)

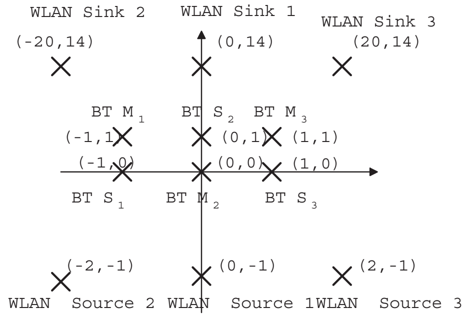

Fig. 4. Topology 2 - Multi-WLANs and Bluetooth piconets interference

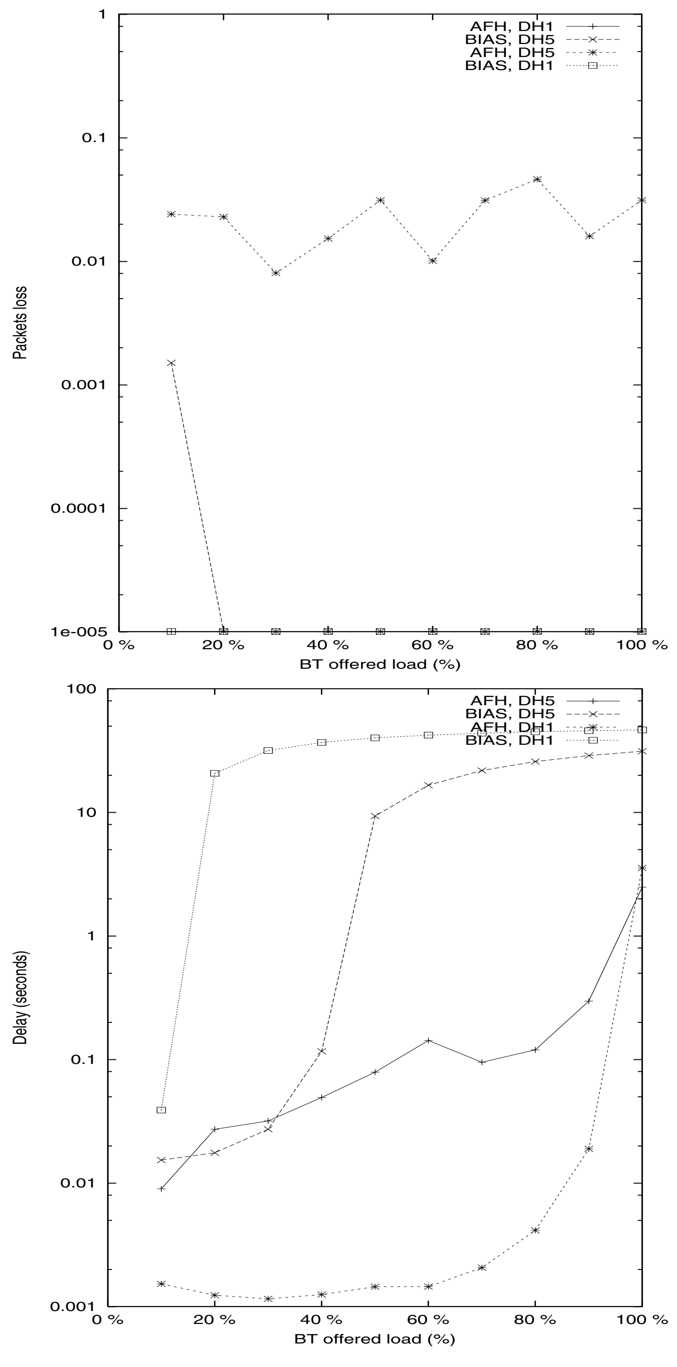

Fig. 5. $\boxed{\frac{(a)}{(b)}}$ Multi-WLANs Interference. (a) Probability of Packet Loss. (b) Mean Access Delay (seconds)

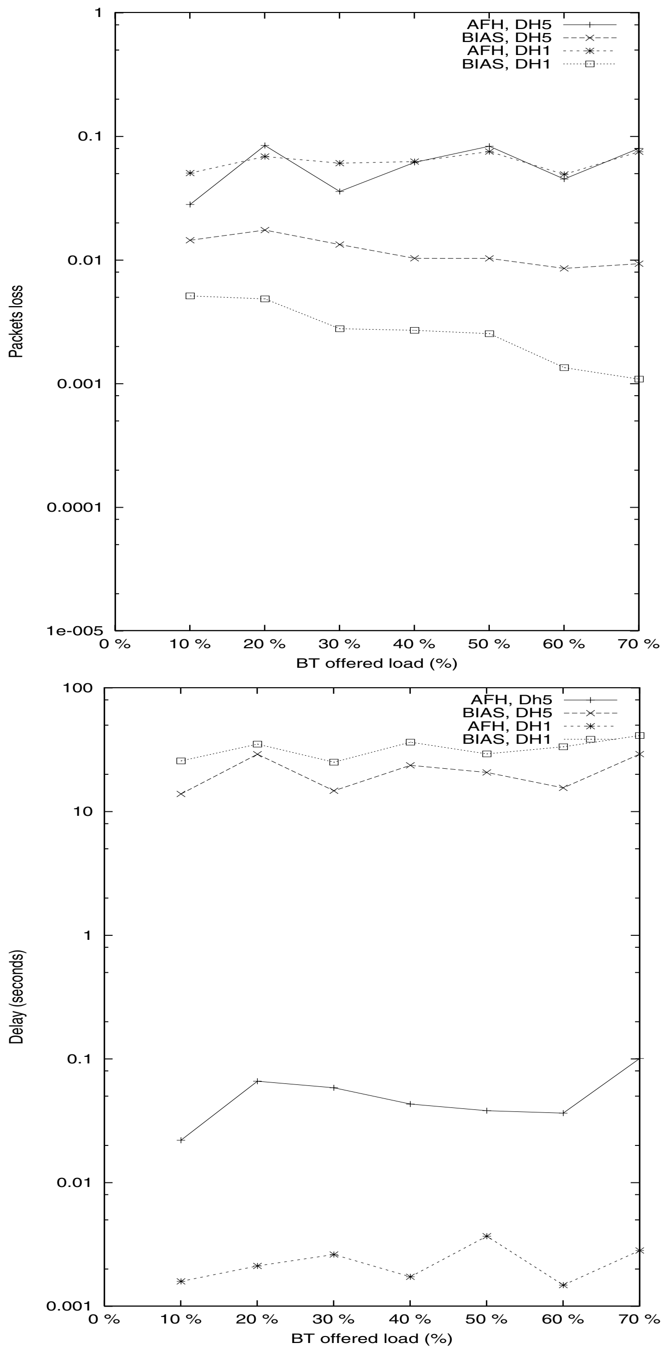

Fig. 6. $\boxed{\frac{(a)}{(b)}}$ Multi-WLANs and Multi-Bluetooth Piconets Interference. (a) Probability of Packet Loss. (b) Mean Access Delay (seconds)